功能：求二维k次NURBS曲线的分子曲线上参数为u的点处的r阶导矢(PX,PY)及分母函数的r阶导数PW。
输入参数：m_aVertex-控制顶点，m_Weights-权因子，m_aNode-节点矢量，都是受保护成员。r-导矢阶数；k -曲线次数；u-参数。
输出参数：(PX,PY)- r阶导矢；PW-r阶导数。当u为内节点值,r>0时,m_KnotMark=0与1分布输出右导矢、左导矢与右导数、左导数。

```
void GetWBPr(int r,int k,double u,double &PX,double &PY,double &PW)
{
	CArray<double,double> tempx,tempy,tempw;
	if(r>k)
	{PX=0;  PY=0;  PW=0;  return;}
	for(int i=m_aNode.GetSize()-m_nTimes-2;i>=0;i--)
	{
		if(u>=m_aNode[i]) break;
	}
	if(u<m_aNode[k]) i=k;
	if((k==1&&r==1&&u!=m_aNode[i])||(m_KnotMark==0&&i==k&&u==m_aNode[i])||m_KnotMark==0)
	{

PX=int((m_aVertex[i].x*m_Weights[i]-m_aVertex[i-1].x*m_Weights[i-1])/(m_aNode[i+1]-m_aNode[i]));

PY=int((m_aVertex[i].y*m_Weights[i]-m_aVertex[i-1].y*m_Weights[i-1])/(m_aNode[i+1]-m_aNode[i]));
		PW=(m_Weights[i]-m_Weights[i-1])/(m_aNode[i+1]-m_aNode[i]);
	}
	if(k==1&&r==1&&m_KnotMark==1)
	{

PX=int((m_aVertex[i-1].x*m_Weights[i-1]-m_aVertex[i-2].x*m_Weights[i-2])/(m_aNode[i]-m_aNode[i-1]));

PY=int((m_aVertex[i-1].y*m_Weights[i-1]-m_aVertex[i-2].y*m_Weights[i-2])/(m_aNode[i]-m_aNode[i-1]));
		PW=(m_Weights[i-1]-m_Weights[i-2])/(m_aNode[i]-m_aNode[i-1]);
	}
	if(k>1&&r==k&&m_KnotMark==1&&u==m_aNode[i]&&i!=k) i=i-1;
	if((k==1&&r==0)||k>1)
	{
		for(int j=i-k;j<=i;j++)
		{
			tempx.Add(double(m_aVertex[j].x*m_Weights[j]));
			tempy.Add(double(m_aVertex[j].y*m_Weights[j]));
			tempw.Add(m_Weights[j]);
		}
		int l=0;
		if(k>1)
		{
			for(int l=1;l<=r;l++)  //本修订版教材(7.7)式第一式的r级递推
			{
				for(j=i-k;j<=i-l;j++)
				{
					double beta=(k-l+1)/(m_aNode[j+k+1]-m_aNode[j+l]);
					int jj=j-(i-k);
					tempx[jj]=beta*(tempx[jj+1]-tempx[jj]);
					tempy[jj]=beta*(tempy[jj+1]-tempy[jj]);
					tempw[jj]=beta*(tempw[jj+1]-tempw[jj]);
				}
			}
		}
		//本修订版教材(7.7c)式德布尔算法,次数已由k降为k-r
		for(l=1;l<=k-r;l++)
		{
			for(j=i-k;j<=i-l-r;j++)
			{
				double alpha,du=m_aNode[j+k+1]-m_aNode[j+r+1];
				if(du==0.0) alpha=0.0;
				else alpha=(u-m_aNode[j+r+1])/du;
				tempx[j-i+k]=(1-alpha)*tempx[j-i+k]+alpha*tempx[j-i+k+1];
				tempy[j-i+k]=(1-alpha)*tempy[j-i+k]+alpha*tempy[j-i+k+1];
				tempw[j-i+k]=(1-alpha)*tempw[j-i+k]+alpha*tempw[j-i+k+1];
				double ww=tempw[j-i+k];
			}
		}
		PX=tempx[0];
		PY=tempy[0];
		PW=tempw[0];
		tempx.RemoveAll();
		tempy.RemoveAll();
		tempw.RemoveAll();
```

```
        }
}
```